VIT® UNIVERSITY
(Estd. u/s 3 of UGC Act 1956)
www.vit.ac.in
Vellore ▪ Chennai
CHENNAI CAMPUS
Vandalur - Kelambakkam Road, Chennai - 600127

CUSTOMER SEGMENTATION

J COMPONENT

A Project Report submitted as part of the course.

CSE 3044 - INFORMATION VISUALIZATION

to

Dr. ARUNKUMAR SIVARAMAN

School of Computer Science and Engineering (SCOPE)

BY

P Lokesh Kanna

GROUP CODE: A7

WINTER 2020 - 21

VIT UNIVERSITY, CHENNAI

VANDALUR KELAMBAKAM ROAD

CHENNAI-600127

ABSTRACT

- Customer Segmentation is one the most important applications of unsupervised learning. Using clustering techniques, companies can identify the several segments of customers allowing them to target the potential user base.
- In this machine learning project, we will make use of **K-means clustering** which is the essential algorithm for clustering unlabeled dataset.
- It is the process of division of customer base into several groups of individuals that share a similarity in different ways that are relevant to marketing such as gender, age, interests, and miscellaneous spending habits.
- This project proposes an integrated novel approach for determining target customers using predictive model and also discover their associative buying patterns using Apriori algorithm. After identification of targeted customers and their associative buying pattern, the business managers take the strategic profitable decisions accordingly.
- The most common ways in which businesses segment their customer base are:
    - Demographic information, such as gender, age, familial and marital status, income, education, and occupation.
    - Geographical information, which differs depending on the scope of the company. For localized businesses, this info might pertain to specific towns or counties. For larger companies, it might mean a customer's city, state, or even country of residence.
    - Psychographics, such as social class, lifestyle, and personality traits.
    - Behavioral data, such as spending and consumption habits, product/service usage, and desired benefits.

INTRODUCTION

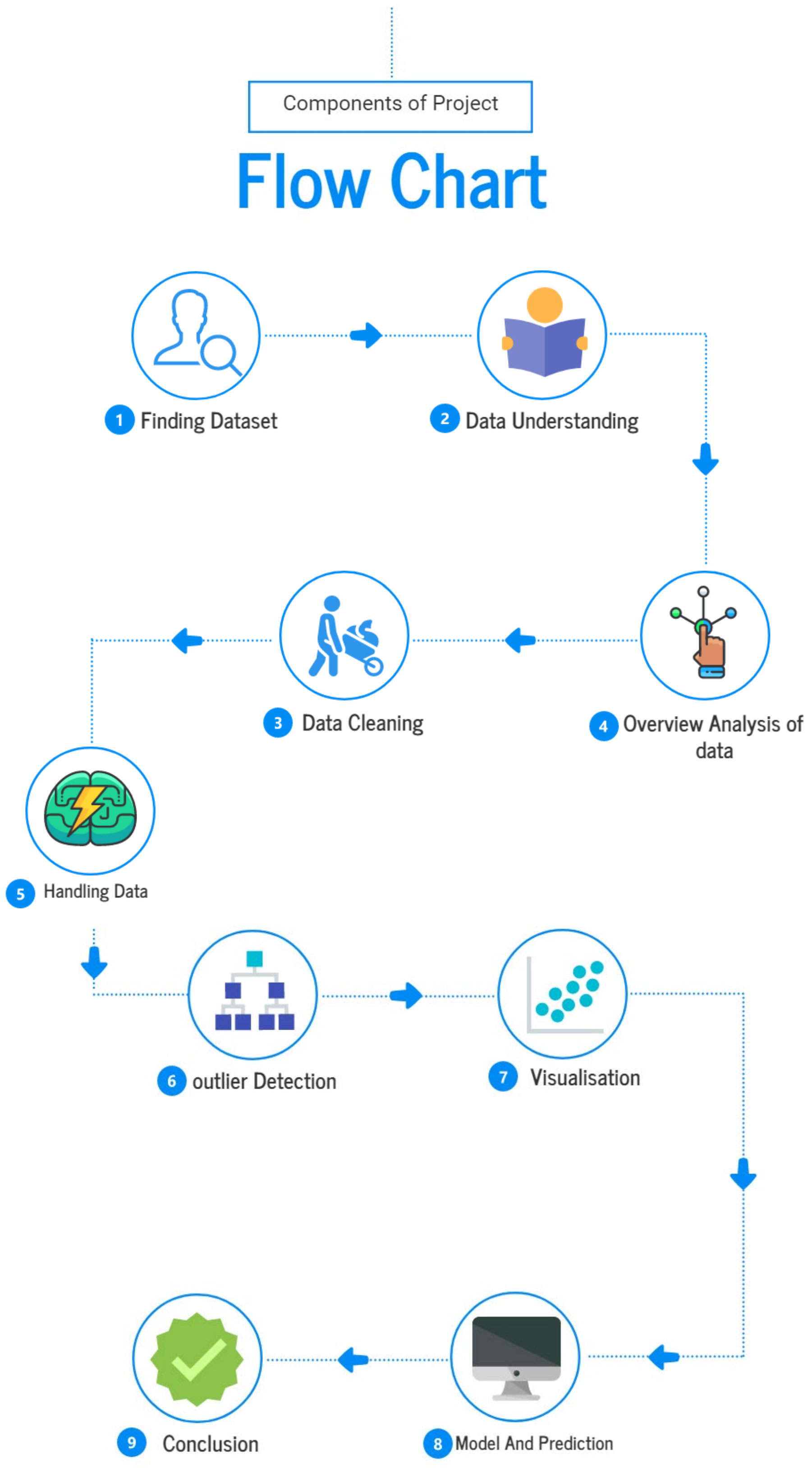
- With the evolution of new technologies and increasing growth of e-commerce it is important for every business to adapt new strategies which help them to win the competitive environment.
- The most valuable asset of any business is customer. In this emerging market it is very difficult to maintain its customer base. To overcome this difficulty every business has to focus on customer segmentation.
- Customer segmentation means categorizing the customers into same group. It acts as a base for Customer Relationship Management (CRM) for businesses to firstly identify their target customers and work on them individually. There are various existing predictive models which provide information on customer segmentation and which help them to segregate the customers.
- For a successful business, apart from retaining and adding new customers, making more profit from each customer is key task. Different variety of models exist which help the business managers to implement the strategic policy according to individual customer taste but each one of it are having their own limitations. To excel the business further this paper is proposing a new integrated approach of customer segmentation combined with association mining of different segmented customer which provides more profit to the business
- Companies that deploy customer segmentation are under the notion that every customer has different requirements and require a specific marketing effort to address them appropriately. Companies aim to gain a deeper approach of the customer they are targeting.
- Therefore, their aim has to be specific and should be tailored to address the requirements of each and every individual customer.
- Furthermore, through the data collected, companies can gain a deeper understanding of customer preferences as well as the requirements for discovering valuable segments that would reap them maximum profit.
- This way, they can strategize their marketing techniques more efficiently and minimize the possibility of risk to their investment.

- The technique of customer segmentation is dependent on several key differentiators that divide customers into groups to be targeted.
- Data related to demographics, geography, economic status as well as behavioral patterns play a crucial role in determining the company direction towards addressing the various segments.

TOOLS AND TECHNOLOGIES USED

- We have used different plots to show insights.
    1. Barplot
    2. Histogram
    3. Pie Chart
    4. Boxplot
    5. Density plot
- K-means Algorithm : There are three popular methods in determining the optimal clusters
    1. Elbow method
       In cluster analysis, the elbow method is a heuristic used in determining the number of clusters in a data set.
       The method consists of plotting the explained variation as a function of the number of clusters, and picking the elbow of the curve as the number of clusters to use.
    2. Silhouette method
       Silhouette refers to a method of interpretation and validation of consistency within clusters of data. The technique provides a succinct graphical representation of how well each object has been classified. The silhouette value is a measure of how similar an object is to its own cluster compared to other clusters.
    3. Gap statistic
       The Gap statistic is a standard method for determining the number of clusters in a set of data. The Gap statistic standardizes the graph of log(Wk), where Wk is the within-cluster dispersion, by comparing it to its expectation under an appropriate null reference distribution of the data.

ARCHITECTURE



DESCRIPTION OF MODULES USED

Library(Plotly)

The plotly Python library is an interactive, open-source plotting library that supports over 40 unique chart types covering a wide range of statistical, financial, geographic, scientific, and 3-dimensional use-cases.

Built on top of the Plotly JavaScript library (plotly.js), plotly enables Python users to create beautiful interactive web-based visualizations that can be displayed in Jupyter notebooks, saved to standalone HTML files, or served as part of pure Python-built web applications using Dash. The plotly Python library is sometimes referred to as "plotly.py" to differentiate it from the JavaScript library.

Thanks to deep integration with the orca image export utility, plotly also provides great support for non-web contexts including desktop editors (e.g. QtConsole, Spyder, PyCharm) and static document publishing (e.g. exporting notebooks to PDF with high-quality vector images).

Library(Purrr)

At its core, R is a functional programming language, and the purrr package enhances the functional programming aspect of R by providing different methods to work with functions and vectors.

The concept of functional programming is a separate topic that needs its own discussion. You can find an overview here, but to give you a basic idea here is a short example.

Suppose you are working with a small data set that only contains numerical values, but due to a data entry error a few columns contain characters. If you were not following functional programming principles, you would go to each cell location to manually replace the data point. If you were following functional programming principles, you would create a function that takes the data set as argument and replaces characters with appropriate values.

Library(Cluster)
The **ClusterR** package consists of centroid-based (k-means, mini-batch-kmeans, k-medoids) and distribution-based (GMM) clustering algorithms. Furthermore, the package offers functions to

- validate the output using the true labels,
- plot the results using either a silhouette or a 2-dimensional plot,
- predict new observations,
- estimate the optimal number of clusters for each algorithm separately.

Library(Nbclust)

- NbClust package provides 30 indices which determine the number of clusters in a data set and it offers also the best clustering scheme from different results to the user.
- In addition, it provides a function to perform k-means and hierarchical clustering with different distance measures and aggregation methods.
- Any combination of validation indices and clustering methods can be requested in a single function call.
- This enables the user to simultaneously evaluate several clustering schemes while varying the number of clusters, to help determining the most appropriate number of clusters for the data set of interest

Library(Factoextra)

- **Factoextra** is an R package making easy to extract and visualize the output of exploratory multivariate data analyses, including:

  1. **Principal Component Analysis (PCA),** which is used to summarize the information contained in a continuous (i.e, quantitative) multivariate data by reducing the dimensionality of the data without loosing important information.

  2. **Correspondence Analysis (CA),** which is an extension of the principal component analysis suited to analyse a large contingency table formed by two qualitative variables (or categorical data).

3. **Multiple Correspondence Analysis (MCA),** which is an adaptation of CA to a data table containing more than two categorical variables.

4. **Multiple Factor Analysis (MFA)** dedicated to datasets where variables are organized into groups (qualitative and/or quantitative variables).

5. **Hierarchical Multiple Factor Analysis (HMFA):** An extension of MFA in a situation where the data are organized into a hierarchical structure.

6. **Factor Analysis of Mixed Data (FAMD),** a particular case of the MFA, dedicated to analyze a data set containing both quantitative and qualitative variables.

Library(Ggplot2)

ggplot2 is a system for declaratively creating graphics, based on The Grammar of Graphics. You provide the data, tell ggplot2 how to map variables to aesthetics, what graphical primitives to use, and it takes care of the details.

It's hard to succinctly describe how ggplot2 works because it embodies a deep philosophy of visualisation. However, in most cases you start with ggplot(), supply a dataset and aesthetic mapping (with aes()). You then add on layers (like geom_point() or geom_histogram()), scales (like scale_colour_brewer()), faceting specifications (like facet_wrap()) and coordinate systems (like coord_flip()).

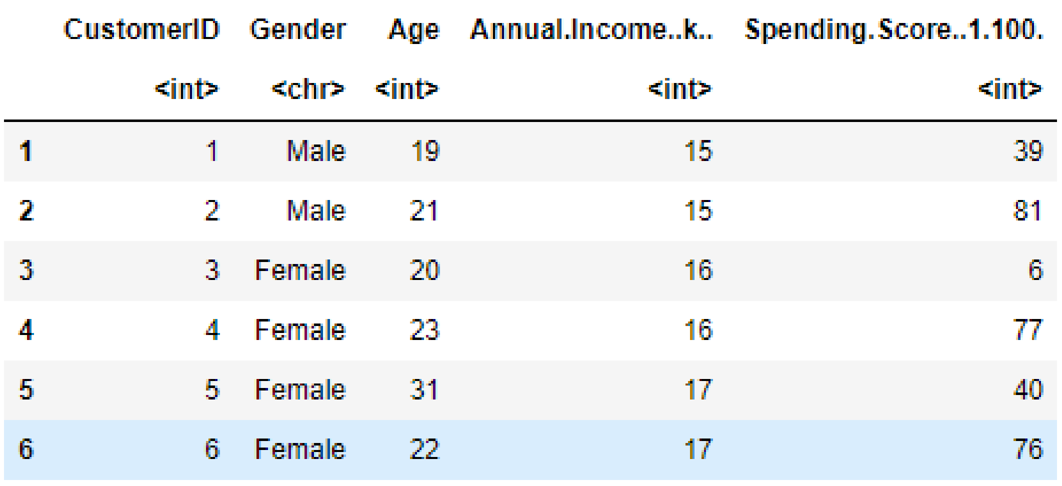
library(DT)

he R package **DT** provides an R interface to the JavaScript library **DataTables**. R data objects (matrices or data frames) can be displayed as tables on HTML pages, and **DataTables** provides filtering, pagination, sorting, and many other features in the tables.

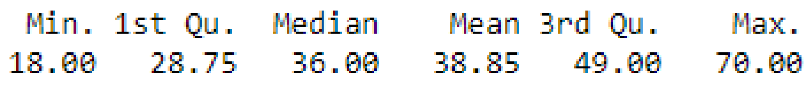
The main function in this package is datatable(), which returns a table widget that can be rendered in R Markdown documents, Shiny apps, and the R console. It is easy to customize the style (cell borders, row striping, and row highlighting, etc), theme (default or Bootstrap), row/column names, table caption, and so on.

RESULTS

DATASET



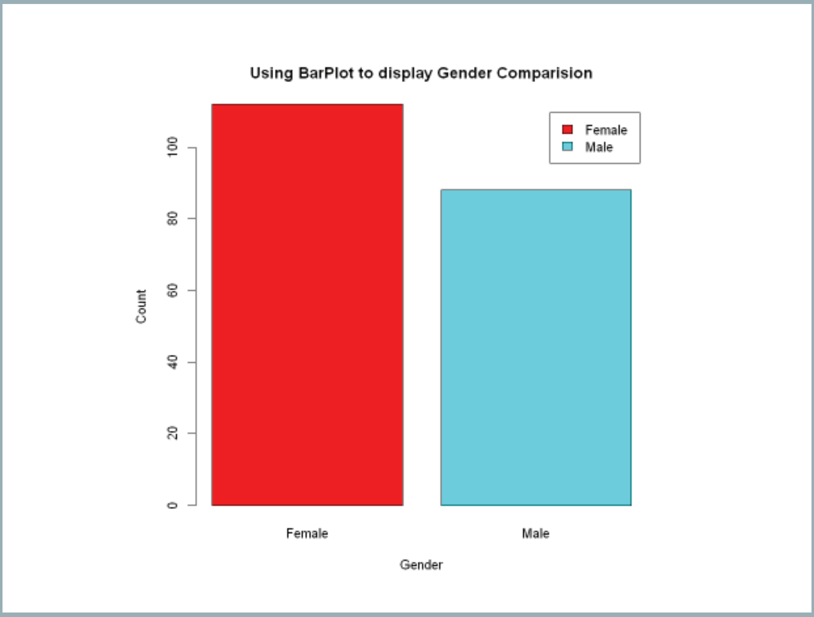
	CustomerID	Gender	Age	Annual.Income..k..	Spending.Score..1.100.
	<int>	<chr>	<int>	<int>	<int>
1	1	Male	19	15	39
2	2	Male	21	15	81
3	3	Female	20	16	6
4	4	Female	23	16	77
5	5	Female	31	17	40
6	6	Female	22	17	76



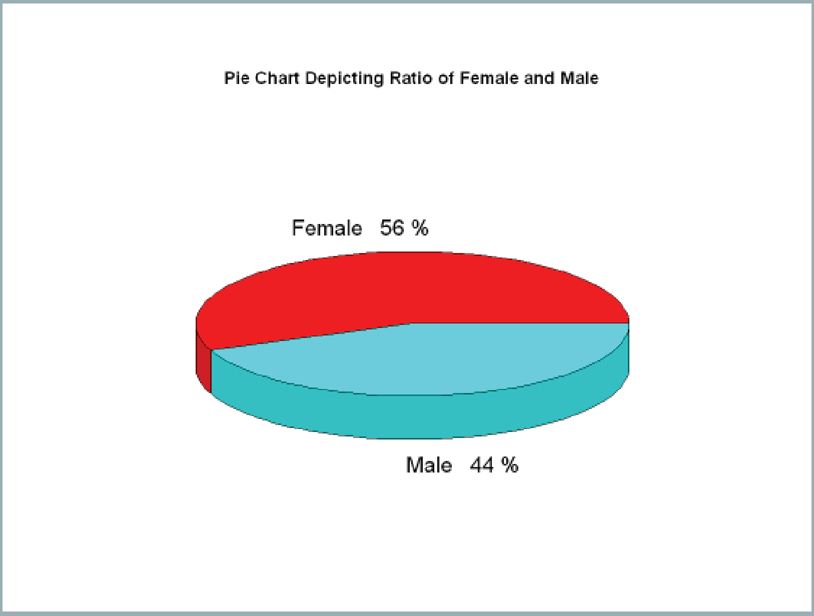
```
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
 18.00   28.75   36.00   38.85   49.00   70.00 
```

OUTPUTS

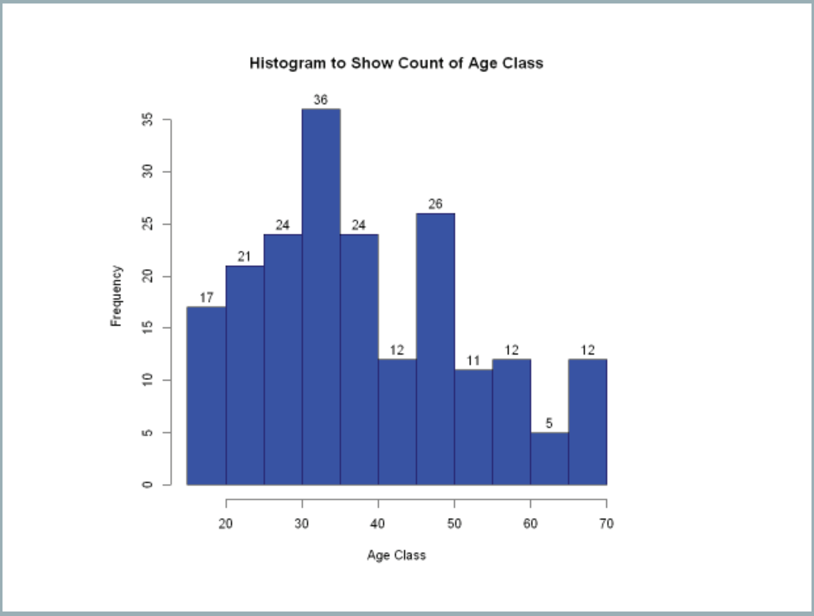
BAR PLOT
GENDER



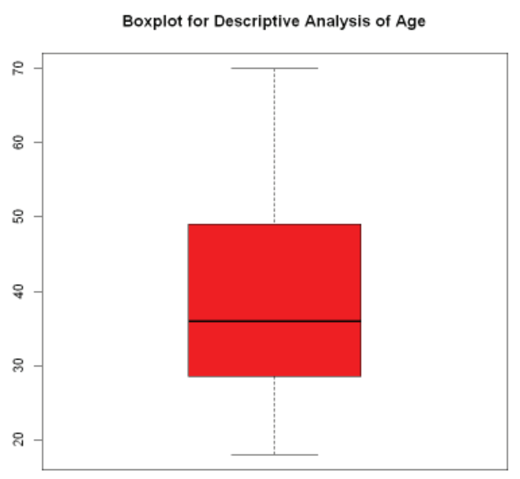
PIE CHART
GENDER



HISTOGRAM - AGE

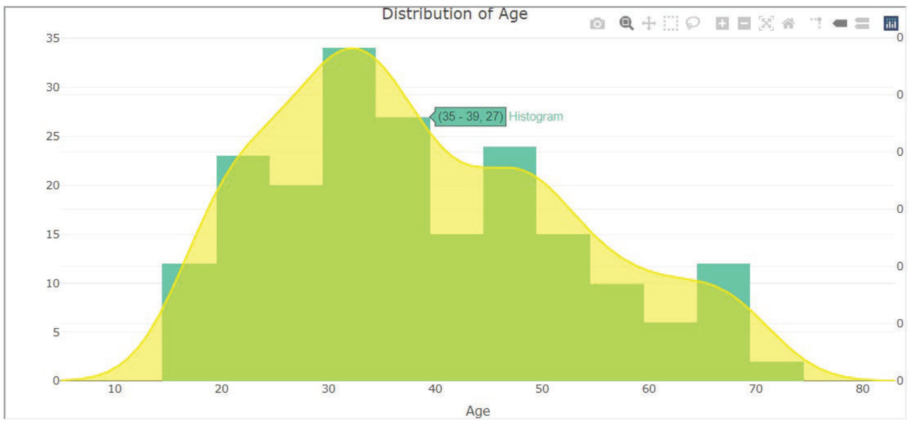


BOX PLOT - AGE

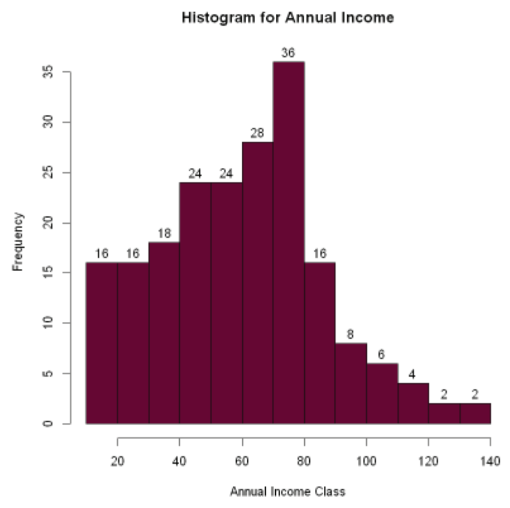


```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  18.00   28.75   36.00   38.85   49.00   70.00 
```

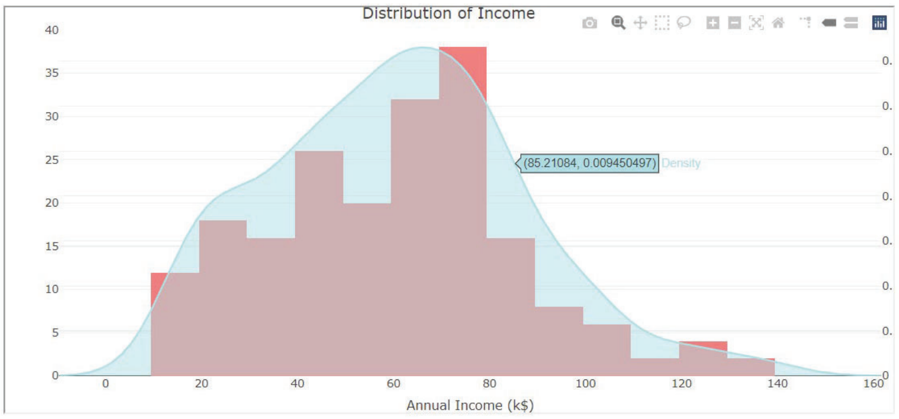
VISUALISATION OF AGE



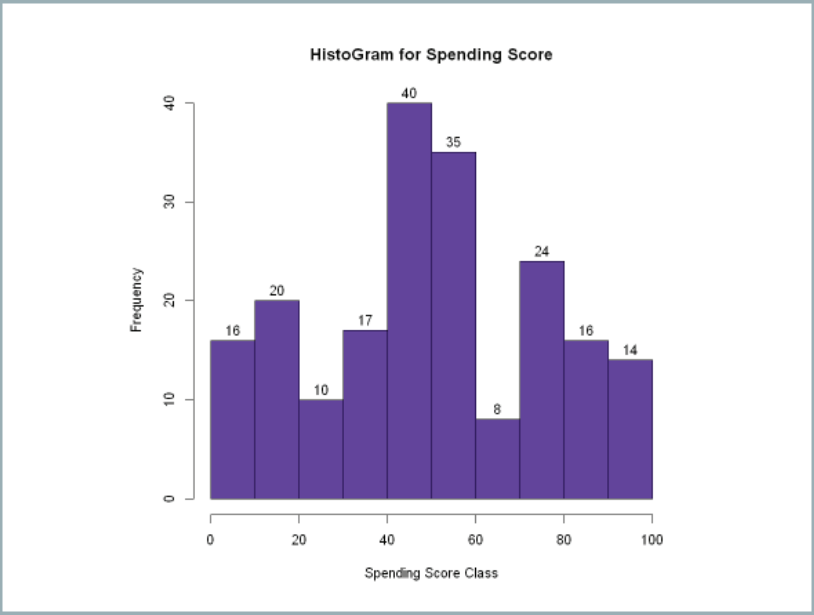
HISTOGRAM
ANNUAL INCOME



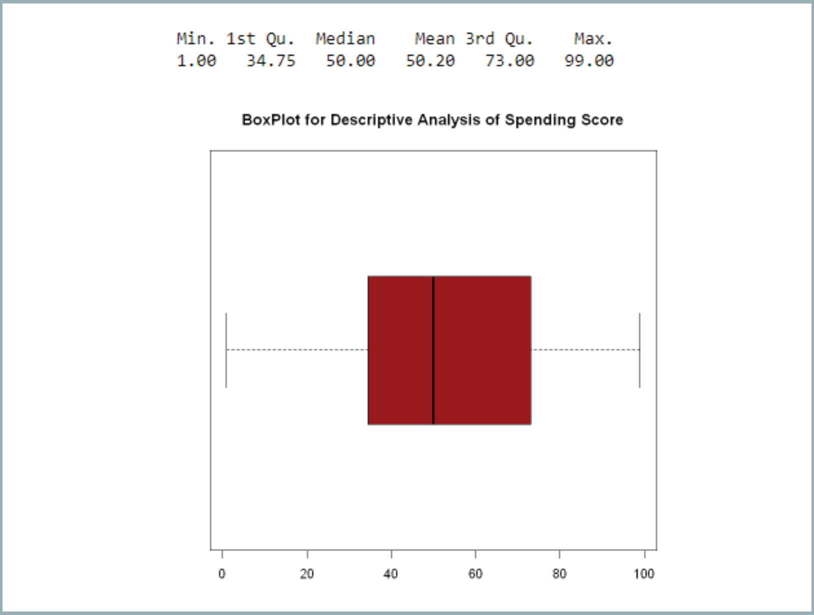
VISUALISATION OF ANNUAL INCOME



HISTOGRAM
SPENDING SCORE

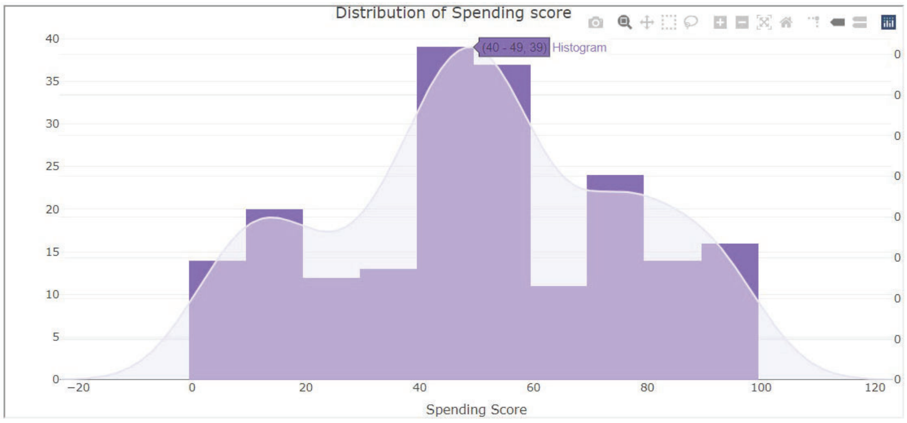


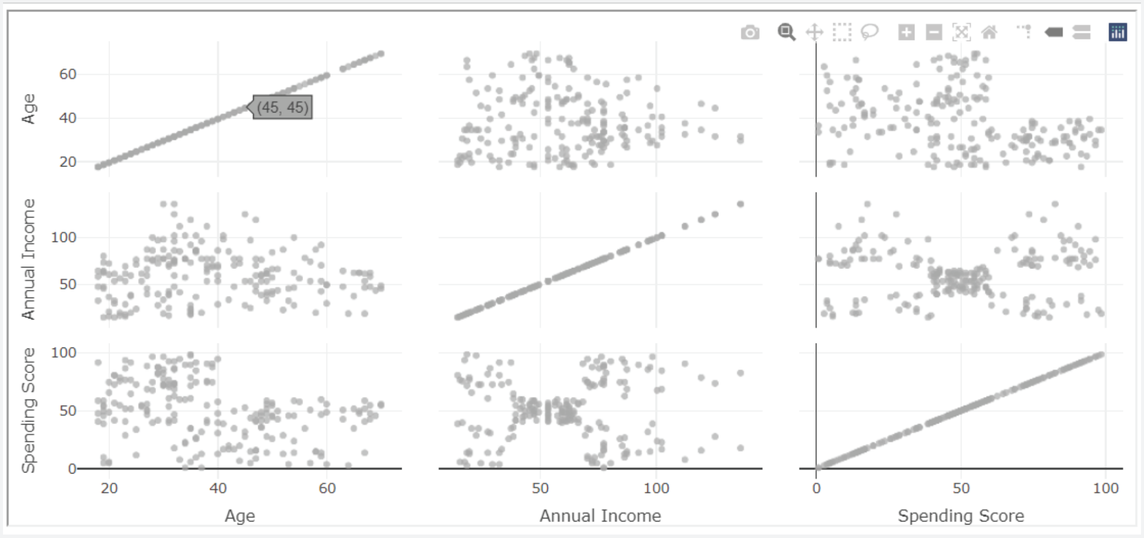
BOX PLOT
SPENDING SCORE



```
Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
1.00   34.75   50.00   50.20   73.00   99.00
```

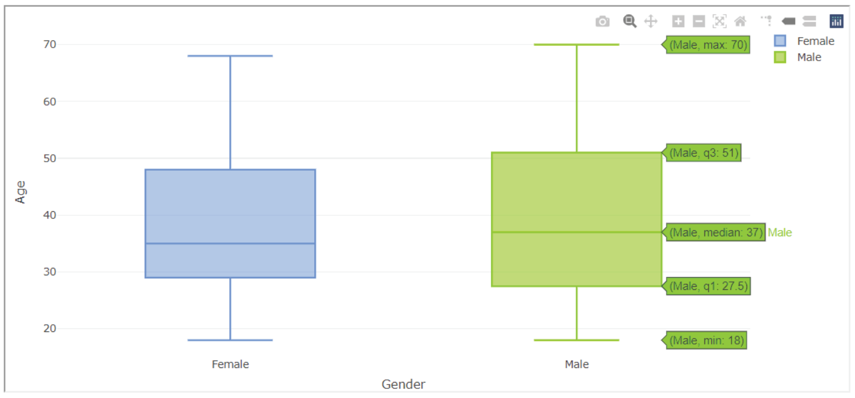
VISUALISATION OF SPENDING SCORE



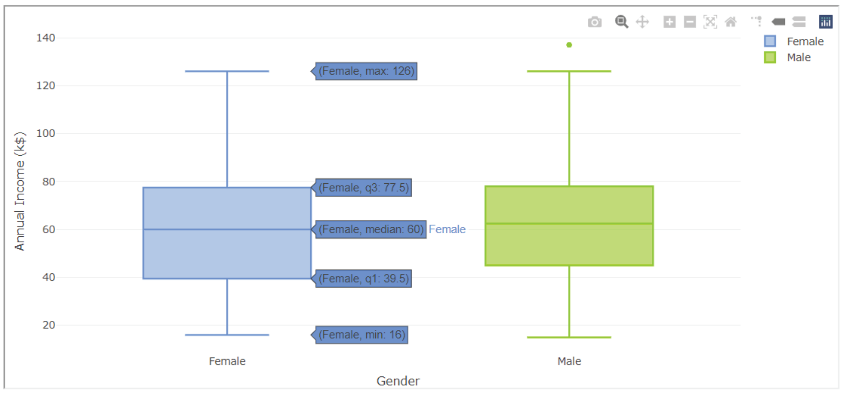


CORRELATION BETWEEEN ATTRIBUTES

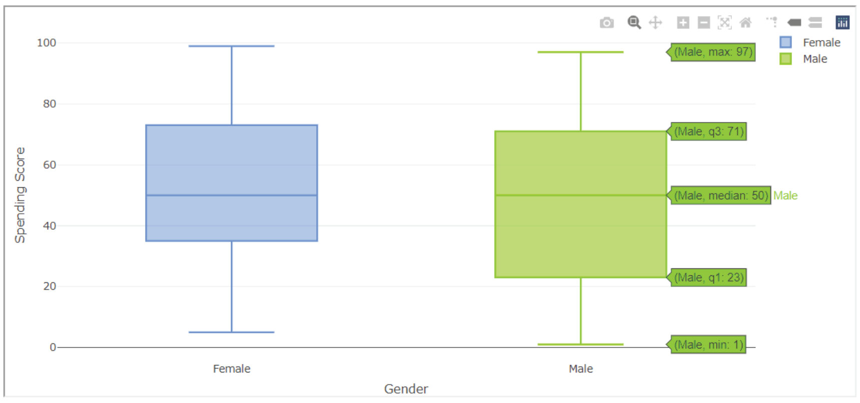
BOX PLOT – AGE(MALE/FEMALE)

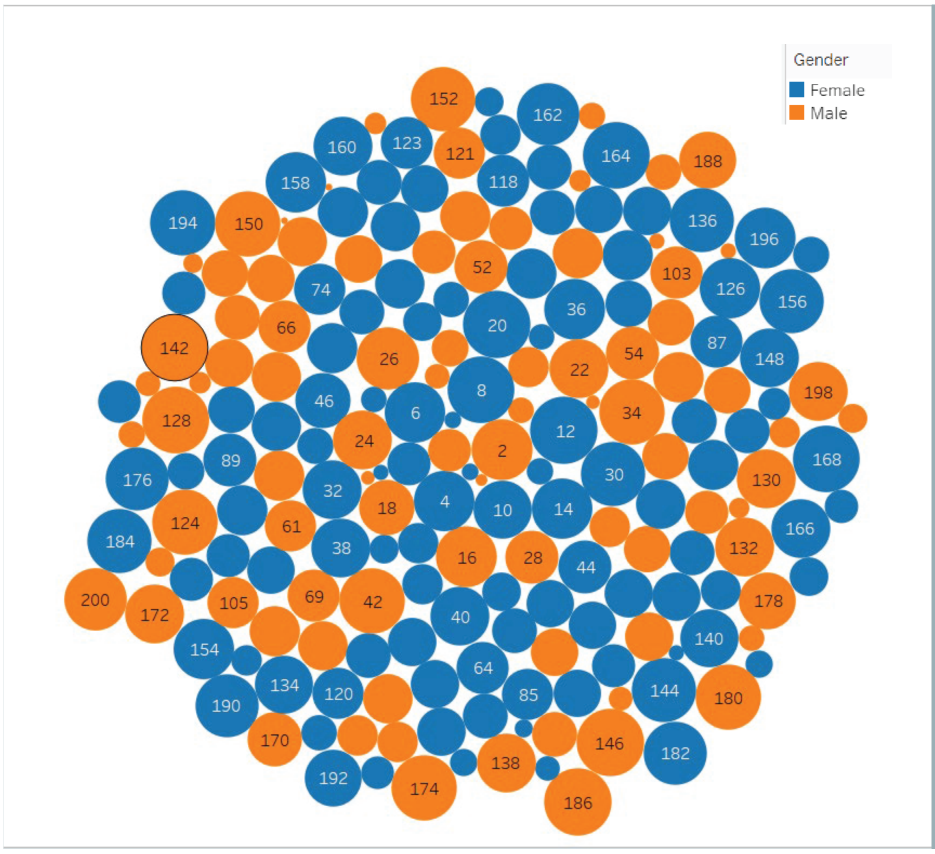


BOX PLOT – ANNUAL INCOME

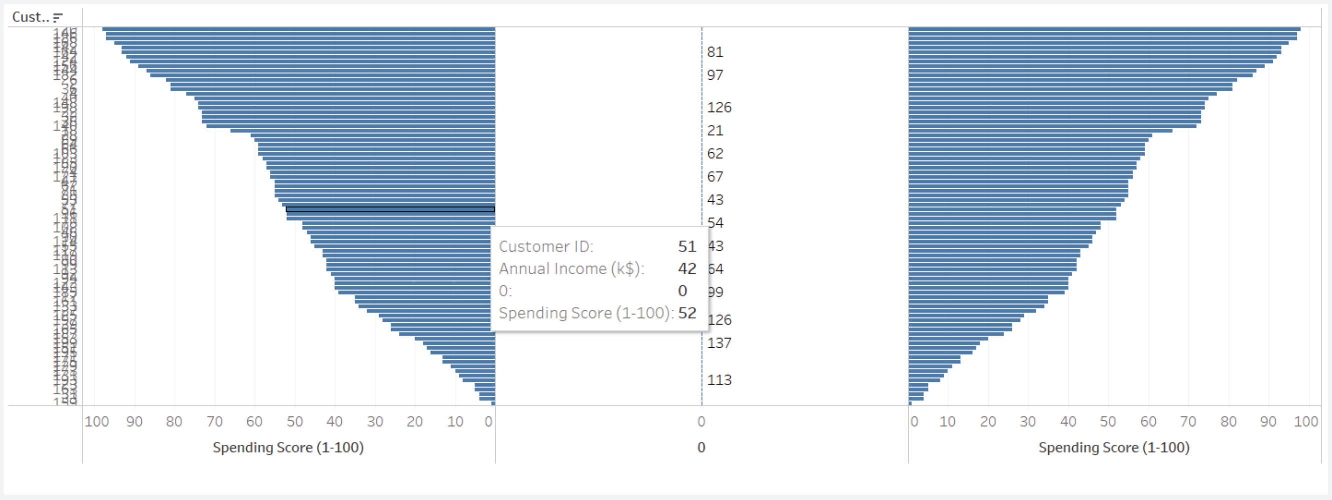


BOX PLOT
SPENDING SCORE(MALE/FEMALE)

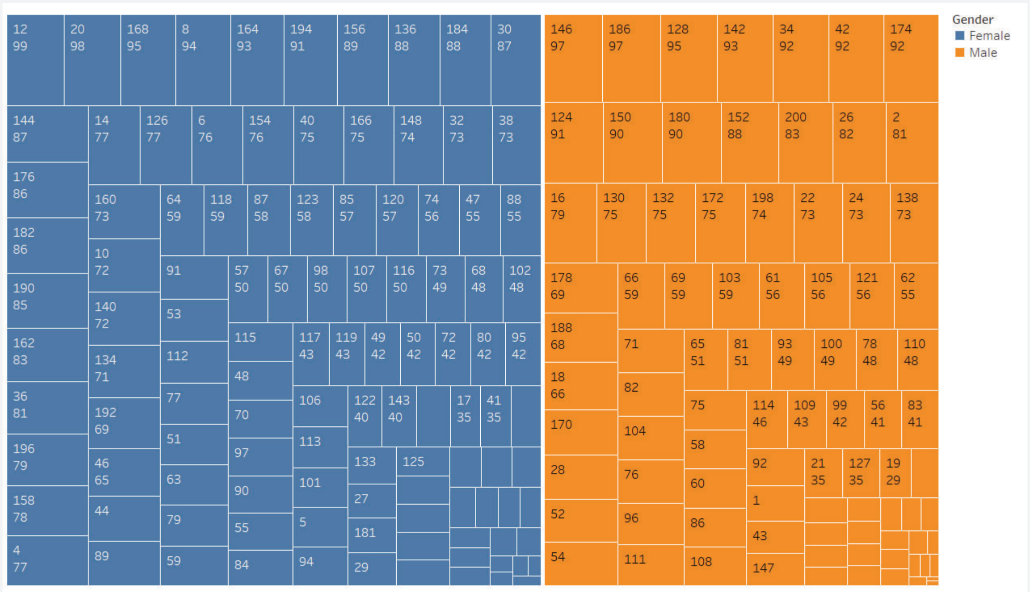




PACKED BUBBLE CHART



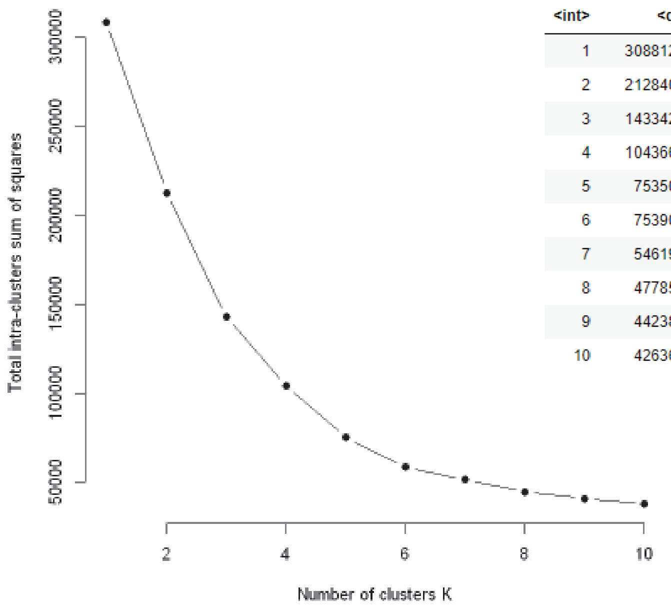
BUTTERFLY CHART

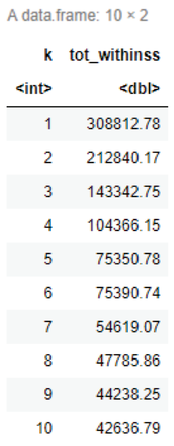


Customer ID and sum of Spending Score (1-100). Color shows details about Gender. Size shows sum of Spending Score (1-100). The marks are labeled by Customer ID and sum of Spending Score (1-100).

TREE GRAPH

DETERMINING THE OPTIMAL NUMBER OF CLUSTERS

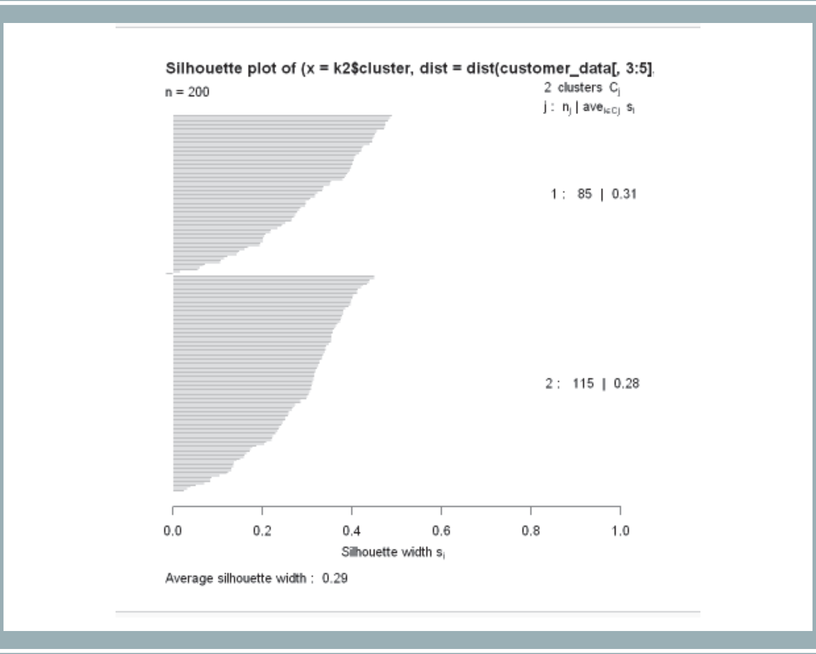


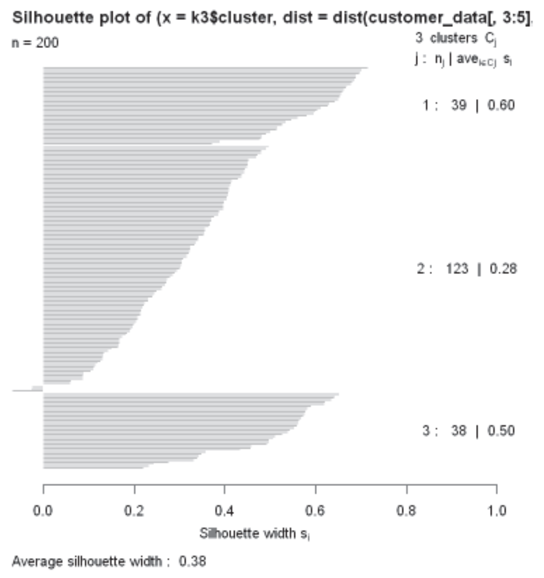

A data.frame: 10 × 2

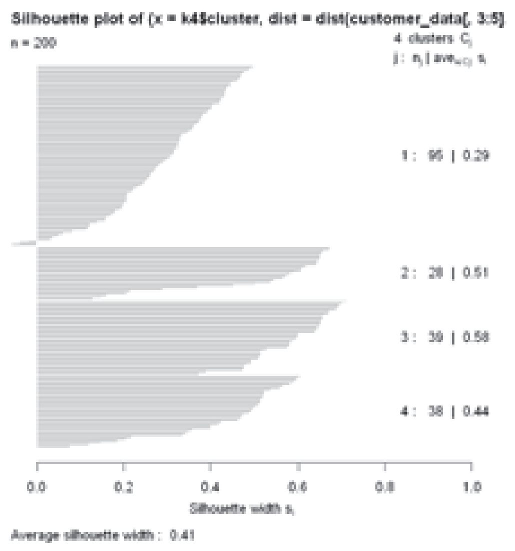
k	tot_withinss
<int>	<dbl>
1	308812.78
2	212840.17
3	143342.75
4	104366.15
5	75350.78
6	75390.74
7	54619.07
8	47785.86
9	44238.25
10	42636.79

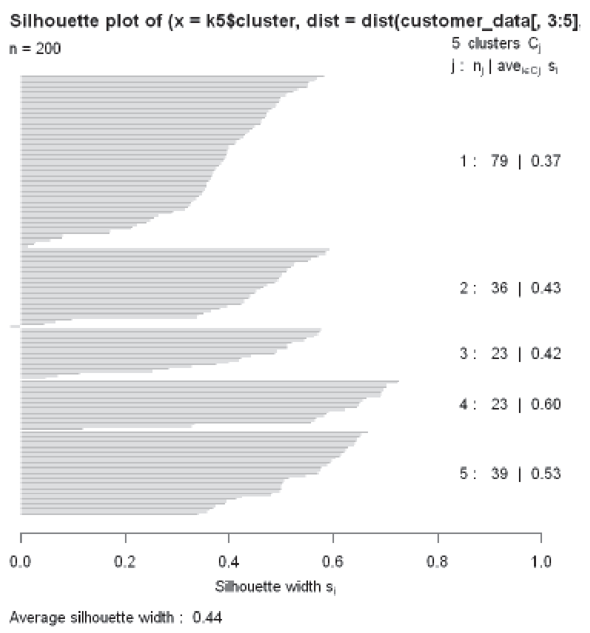
ELBOW METHOD

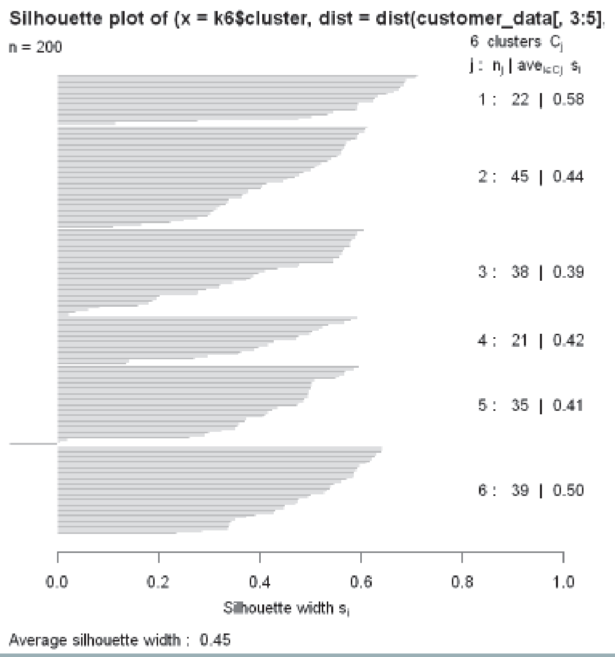
SILHOUETTE METHOD

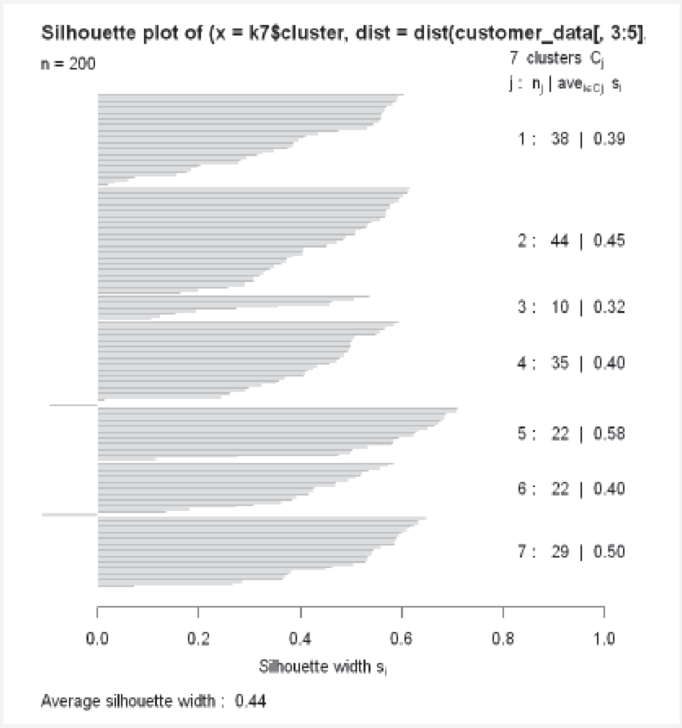


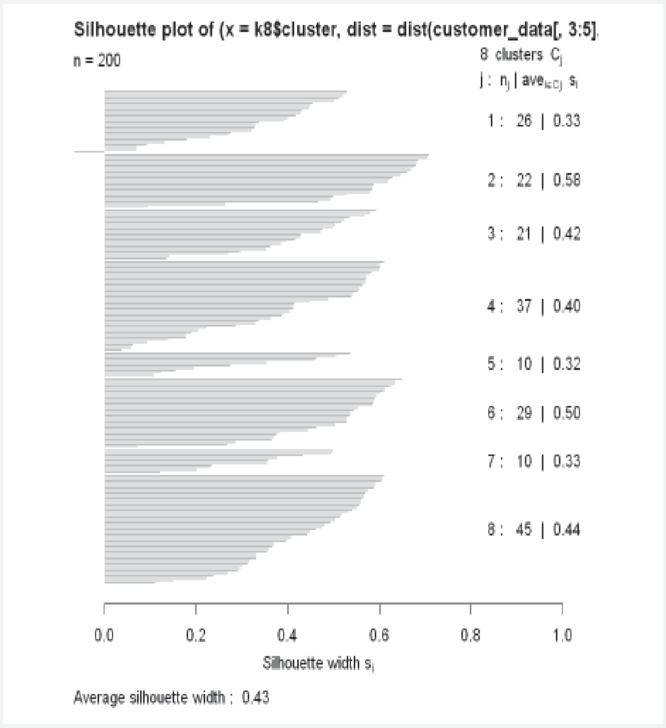

Silhouette plot of (x = k3$cluster, dist = dist(customer_data[, 3:5]
n = 200
3 clusters C_j
j : n_j | $ave_{i\in Cj}$ s_i
1 : 39 | 0.60
2 : 123 | 0.28
3 : 38 | 0.50
0.0 0.2 0.4 0.6 0.8 1.0
Silhouette width s_i
Average silhouette width : 0.38

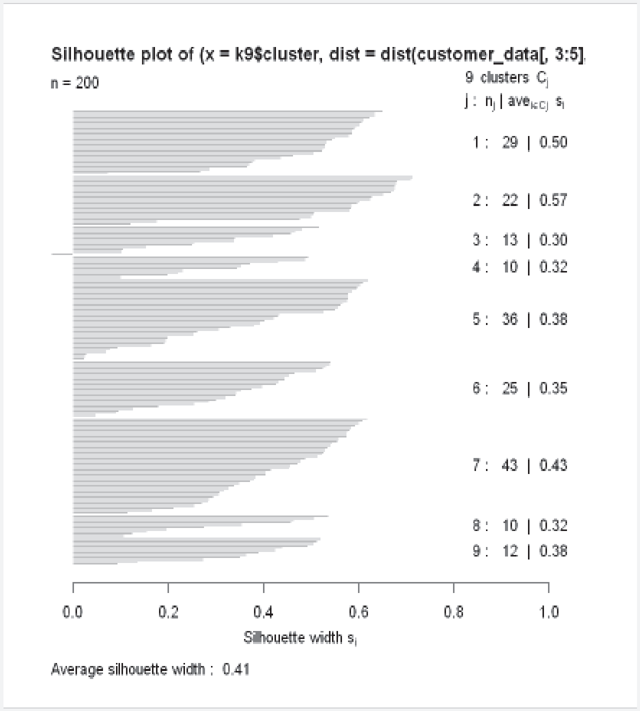

Silhouette plot of (x = k4$cluster, dist = dist(customer_data[, 3:5]
n = 200
4 clusters C_j
j : n_j | $ave_{i\in Cj}$ s_i
1 : 96 | 0.29
2 : 28 | 0.51
3 : 39 | 0.58
4 : 38 | 0.44
0.0 0.2 0.4 0.6 0.8 1.0
Silhouette width s_i
Average silhouette width : 0.41

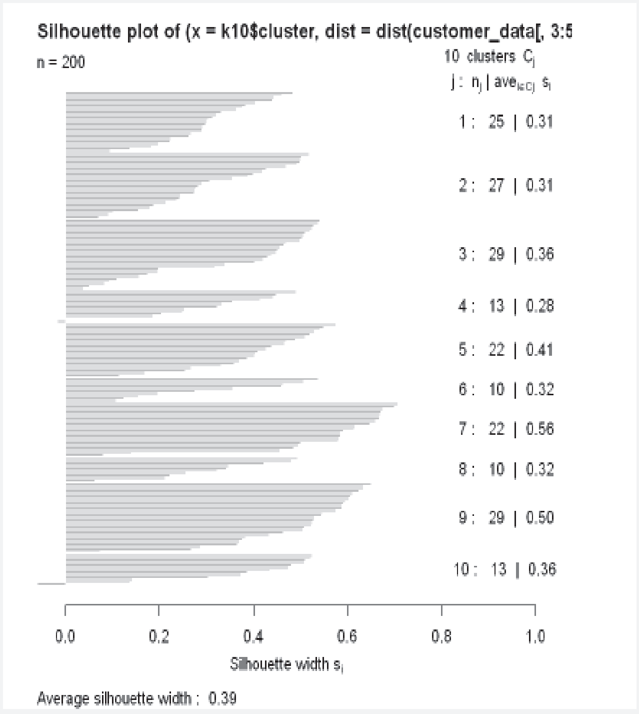

Silhouette plot of (x = k5$cluster, dist = dist(customer_data[, 3:5]
n = 200
5 clusters C_j
j : n_j | $ave_{i\in Cj}$ s_i
1 : 79 | 0.37
2 : 36 | 0.43
3 : 23 | 0.42
4 : 23 | 0.60
5 : 39 | 0.53
0.0 0.2 0.4 0.6 0.8 1.0
Silhouette width s_i
Average silhouette width : 0.44

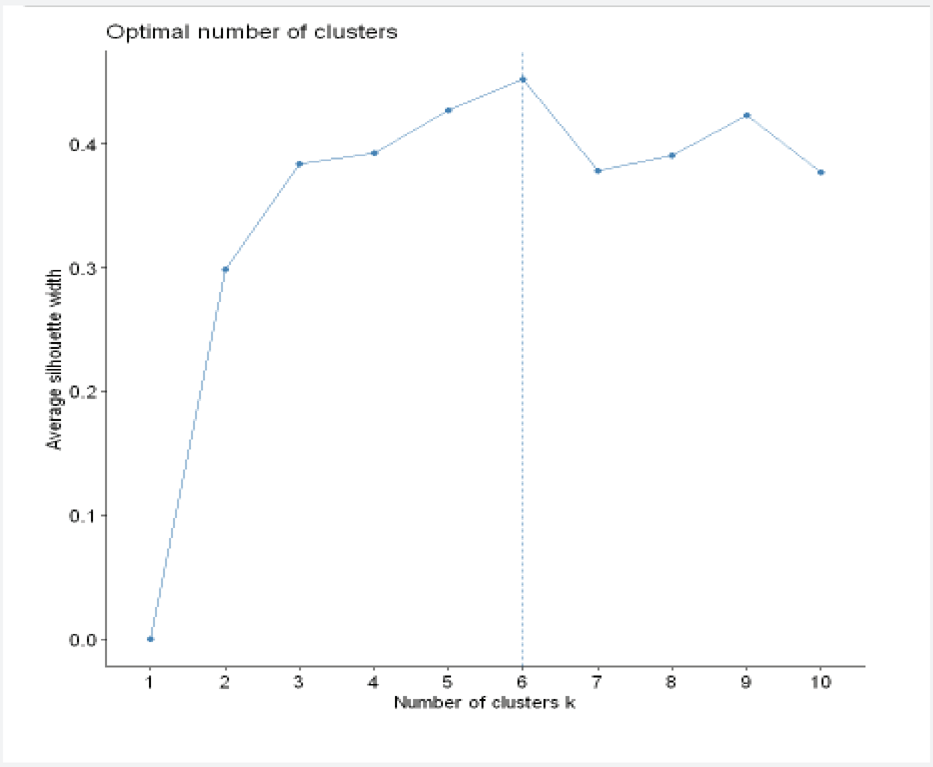

Silhouette plot of (x = k6$cluster, dist = dist(customer_data[, 3:5]
n = 200
6 clusters C_j
j : n_j | $ave_{i\in Cj}$ s_i
1 : 22 | 0.58
2 : 45 | 0.44
3 : 38 | 0.39
4 : 21 | 0.42
5 : 35 | 0.41
6 : 39 | 0.50
0.0 0.2 0.4 0.6 0.8 1.0
Silhouette width s_i
Average silhouette width : 0.45


Silhouette plot of (x = k7$cluster, dist = dist(customer_data[, 3:5]
n = 200
7 clusters C_j
j : n_j | $ave_{i \in Cj}$ s_i
1 : 38 | 0.39
2 : 44 | 0.45
3 : 10 | 0.32
4 : 35 | 0.40
5 : 22 | 0.58
6 : 22 | 0.40
7 : 29 | 0.50
0.0 0.2 0.4 0.6 0.8 1.0
Silhouette width s_i
Average silhouette width : 0.44

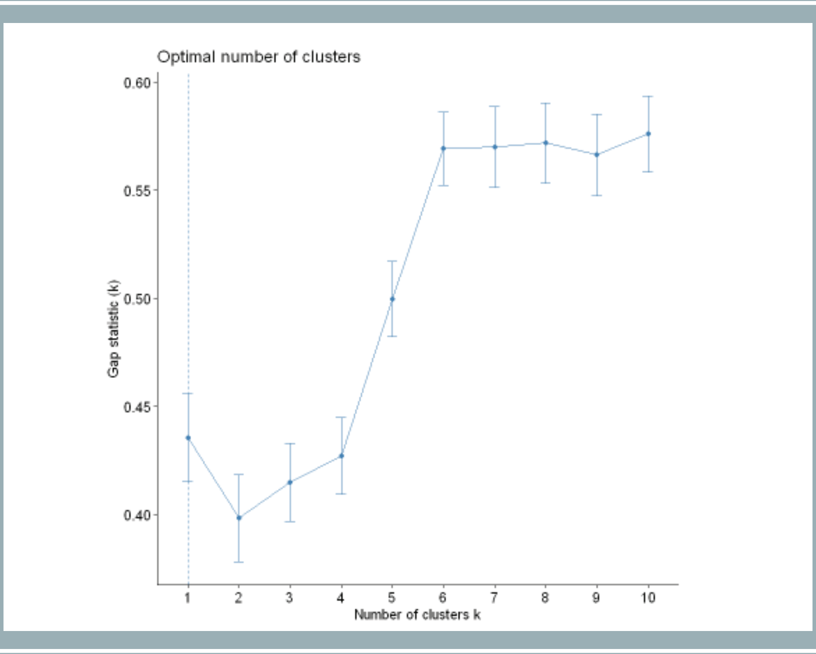

Silhouette plot of (x = k8$cluster, dist = dist(customer_data[, 3:5]
n = 200
8 clusters C_j
j : n_j | $ave_{i \in Cj}$ s_i
1 : 26 | 0.33
2 : 22 | 0.58
3 : 21 | 0.42
4 : 37 | 0.40
5 : 10 | 0.32
6 : 29 | 0.50
7 : 10 | 0.33
8 : 45 | 0.44
0.0 0.2 0.4 0.6 0.8 1.0
Silhouette width s_i
Average silhouette width : 0.43

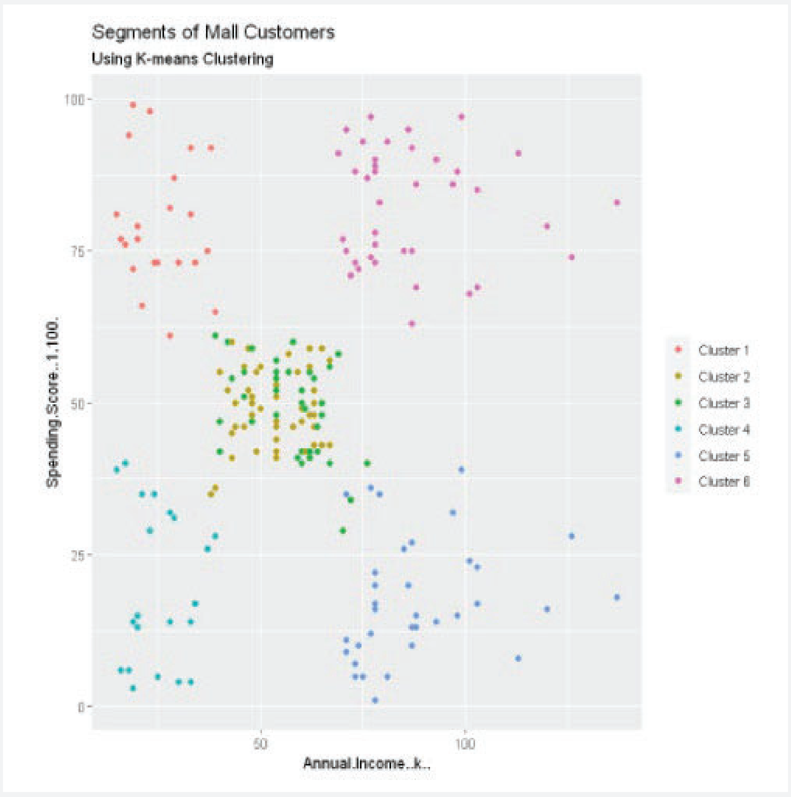

Silhouette plot of (x = k9$cluster, dist = dist(customer_data[, 3:5]
n = 200
9 clusters C_j
j : n_j | $ave_{i \in Cj}$ s_i
1 : 29 | 0.50
2 : 22 | 0.57
3 : 13 | 0.30
4 : 10 | 0.32
5 : 36 | 0.38
6 : 25 | 0.35
7 : 43 | 0.43
8 : 10 | 0.32
9 : 12 | 0.38
0.0 0.2 0.4 0.6 0.8 1.0
Silhouette width s_i
Average silhouette width : 0.41


Silhouette plot of (x = k10$cluster, dist = dist(customer_data[, 3:5
n = 200
10 clusters C_j
j : n_j | $ave_{i \in Cj}$ s_i
1 : 25 | 0.31
2 : 27 | 0.31
3 : 29 | 0.36
4 : 13 | 0.28
5 : 22 | 0.41
6 : 10 | 0.32
7 : 22 | 0.56
8 : 10 | 0.32
9 : 29 | 0.50
10 : 13 | 0.36
0.0 0.2 0.4 0.6 0.8 1.0
Silhouette width s_i
Average silhouette width : 0.39

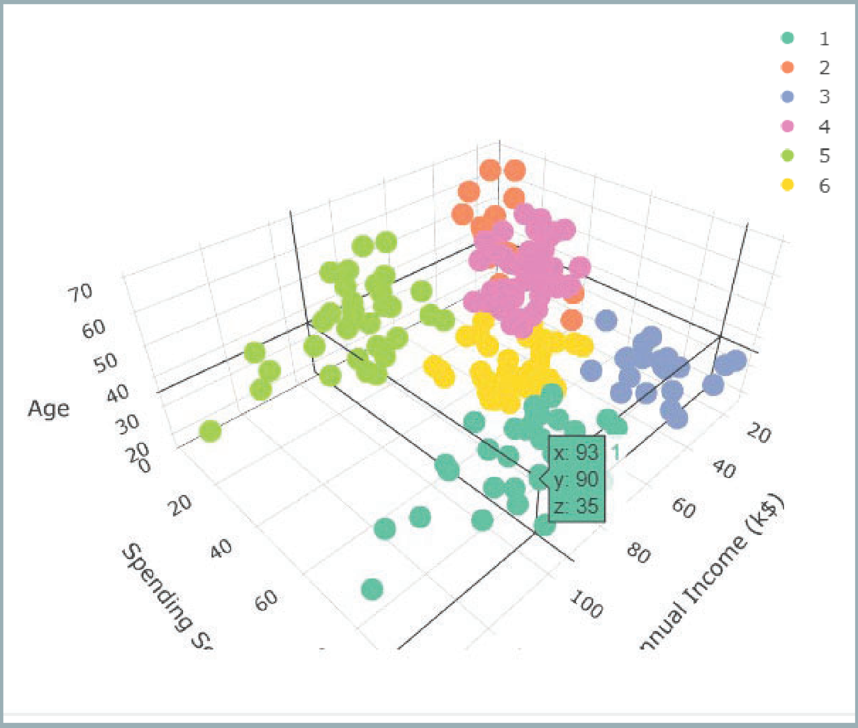


GAP STATISTICS METHOD





SEGMENT
OF
MALL CUSTOMERS
USING
K-MEANS
CLUSTERING



3-D
VISUALISATION
OF
K-MEANS
CLUSTERING

Customer Segmentation

CODE

R LANGUAGE

```
[ ]:
```

```
[ ]: customer_data <- read.csv("Mall_Customers.csv")
     data <- read.csv("Mall_Customers.csv")
```

```
[ ]: library(dplyr)
     library(plotly)
     library(purrr)
     library(cluster)
     library(NbClust)
     library(factoextra)
     library(IRdisplay)
     library(gridExtra)
     library(grid)
     library(plotrix)
     library(foreign)
```

```
[ ]: dim(customer_data)
     str(customer_data)

     names(customer_data)
```

```
[ ]: head(customer_data)
     summary(customer_data$Age)
     sd(customer_data$Age)
```

```
[ ]: summary(customer_data$Annual.Income..k..)
     sd(customer_data$Annual.Income..k..)
```

```
[ ]: summary(customer_data$Spending.Score..1.100.)
     sd(customer_data$Spending.Score..1.100.)
```

1 Customer Gender Visualization

```
[ ]: a=table(customer_data$Gender)
     barplot(a,main="Using BarPlot to display Gender Comparision",
            ylab="Count",
       xlab="Gender",
       col=rainbow(2),
       legend=rownames(a))
```

[ ]:
```
pct=round(a/sum(a)*100)
lbs=paste(c("Female","Male")," ",pct,"%",sep=" ")
library(plotrix)
pie3D(a,labels=lbs,
   main="Pie Chart Depicting Ratio of Female and Male")
```

[ ]:
```

```

2 Visualization of Age Distribution

[ ]:
```
summary(customer_data$Age)
```

[ ]:
```
hist(customer_data$Age,
    col="blue",
    main="Histogram to Show Count of Age Class",
    xlab="Age Class",
    ylab="Frequency",
    labels=TRUE)
```

[ ]:
```
boxplot(customer_data$Age,
       col="red",
       main="Boxplot for Descriptive Analysis of Age")
```

3 Analysis of the Annual Income of the Customers

[ ]:
```
summary(customer_data$Annual.Income..k..)
hist(customer_data$Annual.Income..k..,
  col="#660033",
  main="Histogram for Annual Income",
  xlab="Annual Income Class",
  ylab="Frequency",
  labels=TRUE)
```

[ ]:
```
plot(density(customer_data$Annual.Income..k..),
    col="yellow",
    main="Density Plot for Annual Income",
    xlab="Annual Income Class",
    ylab="Density")
polygon(density(customer_data$Annual.Income..k..),
        col="#ccff66")
```

4 Analyzing Spending Score of the Customers

[ ]:
```
summary(customer_data$Spending.Score..1.100.)

boxplot(customer_data$Spending.Score..1.100.,
   horizontal=TRUE,
   col="#990000",
   main="BoxPlot for Descriptive Analysis of Spending Score")
```

[ ]:
```
hist(customer_data$Spending.Score..1.100.,
    main="HistoGram for Spending Score",
    xlab="Spending Score Class",
    ylab="Frequency",
    col="#6600cc",
    labels=TRUE)
```

[ ]:
```

```

4.1 Visualization of Age Distribution

[ ]:
```
summary(data$Age)
```

[ ]:
```
density1 <- density(data$Age)
p_age <- data %>% plot_ly(x=~Age) %>%
  add_histogram(color=I("mediumaquamarine"), name = "Histogram") %>%
  add_lines(x = density1$x, y = density1$y, fill = "tozeroy", color =␣
 ↪I("yellow"), yaxis = "y2", name = "Density") %>%
  layout(title = "Distribution of Age ", xaxis = list (title = "Age"),
         yaxis2 = list(overlaying = "y", side = "right"), showlegend = FALSE)
htmlwidgets::saveWidget(p_age, "p_age.html")
display_html('<iframe src="p_age.html" width=100% height=450></iframe>')
```

4.2 Visualization of Annual Income

[ ]:
```
summary(data$Annual.Income..k..)
```

[ ]:
```
density2 <- density(data$Annual.Income..k..)
p_income <- data %>% plot_ly(x=~Annual.Income..k..) %>%
  add_histogram(color=I("lightcoral"), name = "Histogram") %>%
  add_lines(x = density2$x, y = density2$y, fill = "tozeroy", color =␣
 ↪I("powderblue"),
            yaxis = "y2", name="Density") %>%
  layout(title = "Distribution of Income ", xaxis = list (title = "Annual␣
 ↪Income (k$)"),
         yaxis2 = list(overlaying = "y", side = "right"), showlegend = FALSE)
htmlwidgets::saveWidget(p_income, "p_income.html")
display_html('<iframe src="p_income.html" width=100% height=450></iframe>')
```

4.3 Visualization of Spending score

```
[ ]: density3<- density(data$Spending.Score..1.100.)
     p_score <- data %>% plot_ly(x=~Spending.Score..1.100.) %>%
       add_histogram(color=I("mediumpurple"), name="Histogram") %>%
       add_lines(x = density3$x, y = density3$y, fill = "tozeroy", color =␣
      ↪I("lavender"), yaxis = "y2", name="Density") %>%
       layout(title = "Distribution of Spending score ", xaxis = list (title =␣
      ↪"Spending Score"),
              yaxis2 = list(overlaying = "y", side = "right"), showlegend = FALSE)
     htmlwidgets::saveWidget(p_score, "p_score.html")
     display_html('<iframe src="p_score.html" width=100% height=450></iframe>')
```

```
[ ]:
```

```
[ ]:
```

5 Correlation visualization

```
[ ]: p_splom <- customer_data %>%
       plot_ly() %>%
       add_trace(
         type = "splom",
         dimensions = list(
           list(label="Age", values=~Age),
           list(label="Annual Income", values=~Annual.Income..k..),
           list(label="Spending Score", values=~Spending.Score..1.100.)),
         color = I("darkgrey"), opacity = 0.7)
     htmlwidgets::saveWidget(p_splom, "p_splom.html")
     display_html('<iframe src="p_splom.html" width=100% height=450></iframe>')
```

```
[ ]: p1 <- data %>% plot_ly(x=~Gender, y=~Age, color = ~Gender) %>%
       add_boxplot(colors = c("cornflowerblue", "lawngreen"))
     p2 <- data %>% plot_ly(x=~Gender, y=~Annual.Income..k.., color = ~Gender) %>%
       add_boxplot(colors = c("cornflowerblue", "lawngreen")) %>%
       layout(yaxis = list(title ="Annual Income (k$)"))
     p3 <- data %>% plot_ly(x=~Gender, y=~Spending.Score..1.100., color = ~Gender)␣
      ↪%>%
       add_boxplot(colors = c("cornflowerblue", "lawngreen")) %>%
       layout(yaxis = list(title ="Spending Score"))
     htmlwidgets::saveWidget(p1, "p1.html")
     display_html('<iframe src="p1.html" width=100% height=450></iframe>')
     htmlwidgets::saveWidget(p2, "p2.html")
     display_html('<iframe src="p2.html" width=100% height=450></iframe>')
     htmlwidgets::saveWidget(p3, "p3.html")
     display_html('<iframe src="p3.html" width=100% height=450></iframe>')
```

6 Clustering data

1.K-means Algorithm

7 The Elbow method

[ ]:
```
tot_withinss <- map_dbl(1:10, function(k){
  model <- kmeans(data[,3:5], centers = k)
  model$tot.withinss #total within-cluster sum of squares
})

#Create elbow data frame
elbow_df <- data.frame(k= 1:10,
                       tot_withinss = tot_withinss)
elbow_df
```

[ ]:
```
# Visualize the elbow data frame
p_elbow_df <- elbow_df %>%  plot_ly(x=~k, y=~tot_withinss, type="scatter",␣
 ↪mode="markers+lines") %>%
  layout(xaxis = list(dtick = 1))
htmlwidgets::saveWidget(p_elbow_df, "p_elbow_df.html")
display_html('<iframe src="p_elbow_df.html" width=100% height=450></iframe>')
```

[ ]:
```
library(purrr)
set.seed(123)
# function to calculate total intra-cluster sum of square
iss <- function(k) {
  kmeans(customer_data[,3:5],k,iter.max=100,nstart=100,algorithm="Lloyd" )$tot.
 ↪withinss
}

k.values <- 1:10


iss_values <- map_dbl(k.values, iss)

plot(k.values, iss_values,
    type="b", pch = 19, frame = FALSE,
    xlab="Number of clusters K",
    ylab="Total intra-clusters sum of squares")
```

8 The Silhoutte method

[ ]:
```
#Use map_dbl to run models with varying value of k
sil_width <- map_dbl(2:10, function(k){
  model <- pam(x=data[,3:5], k=k)
  model$silinfo$avg.width
})
```

[ ]:
```
# Generate a data frame containing both k and sil_width
sil_df <- data.frame( k= 2:10,
                      sil_width = sil_width)
sil_df
```

[ ]:
```
#Visualize the Silhouette data frame
p_sil_df <- sil_df %>% plot_ly(x=~k, y=~sil_width, type="scatter",␣
 ↪mode="markers+lines") %>%
  layout(xaxis = list(dtick = 1))
htmlwidgets::saveWidget(p_sil_df, "p_sil_df.html")
display_html('<iframe src="p_sil_df.html" width=100% height=450></iframe>')
```

8.1 Average Silhouette Method

[ ]:
```
k2<-kmeans(customer_data[,3:5],2,iter.max=100,nstart=50,algorithm="Lloyd")
s2<-plot(silhouette(k2$cluster,dist(customer_data[,3:5],"euclidean")))
```

[ ]:
```
k3<-kmeans(customer_data[,3:5],3,iter.max=100,nstart=50,algorithm="Lloyd")
s3<-plot(silhouette(k3$cluster,dist(customer_data[,3:5],"euclidean")))
```

[ ]:
```
k4<-kmeans(customer_data[,3:5],4,iter.max=100,nstart=50,algorithm="Lloyd")
s4<-plot(silhouette(k4$cluster,dist(customer_data[,3:5],"euclidean")))
```

[ ]:
```
k5<-kmeans(customer_data[,3:5],5,iter.max=100,nstart=50,algorithm="Lloyd")
s5<-plot(silhouette(k5$cluster,dist(customer_data[,3:5],"euclidean")))
```

[ ]:
```
k6<-kmeans(customer_data[,3:5],6,iter.max=100,nstart=50,algorithm="Lloyd")
s6<-plot(silhouette(k6$cluster,dist(customer_data[,3:5],"euclidean")))
```

[ ]:
```
k7<-kmeans(customer_data[,3:5],7,iter.max=100,nstart=50,algorithm="Lloyd")
s7<-plot(silhouette(k7$cluster,dist(customer_data[,3:5],"euclidean")))
```

[ ]:
```
k8<-kmeans(customer_data[,3:5],8,iter.max=100,nstart=50,algorithm="Lloyd")
s8<-plot(silhouette(k8$cluster,dist(customer_data[,3:5],"euclidean")))
```

[ ]:
```
k9<-kmeans(customer_data[,3:5],9,iter.max=100,nstart=50,algorithm="Lloyd")
s9<-plot(silhouette(k9$cluster,dist(customer_data[,3:5],"euclidean")))
```

[ ]:
```
k10<-kmeans(customer_data[,3:5],10,iter.max=100,nstart=50,algorithm="Lloyd")
s10<-plot(silhouette(k10$cluster,dist(customer_data[,3:5],"euclidean")))
```

[ ]:
```
library(NbClust)
library(factoextra)
```

```
fviz_nbclust(customer_data[,3:5], kmeans, method = "silhouette")
```

9 Gap Statistic Method

```
[ ]: # Use the clusGap function to apply the Gap Statistic Method
gap_stat <- clusGap(data[, 3:5], kmeans, nstart = 25, K.max = 10, B = 50)
print(gap_stat, method = "firstmax")
```

```
[ ]: set.seed(125)
stat_gap <- clusGap(customer_data[,3:5], FUN = kmeans, nstart = 25,
            K.max = 10, B = 50)
fviz_gap_stat(stat_gap)
```

10 Clusters

```
[ ]: set.seed(50)
#Build a k-means model for data with k = 6
model_customers <- kmeans(data[,3:5], centers = 6)
model_customers
```

```
[ ]: #Extract the vector of cluster assignments from the model
clusters <- model_customers$cluster
```

```
[ ]: # Visualizations of 6 clusters
p_clusters <- data %>% plot_ly(x=~Annual.Income..k.., y =~Spending.Score..1.100.
 ↪, z=~Age) %>%
  add_markers(color = factor(clusters)) %>%
  layout(scene = list(
    xaxis = list(title="Annual Income (k$)"),
      yaxis = list(title="Spending Score"),
      zaxis = list(title="Age")))
htmlwidgets::saveWidget(p_clusters, "p_clusters.html")
display_html('<iframe src="p_clusters.html" width=100% height=450></iframe>')
```

10.1 Visualizing the Clustering Results using the First Two Principle Components

```
[ ]: pcclust=prcomp(customer_data[,3:5],scale=FALSE) #principal component analysis
summary(pcclust)

pcclust$rotation[,1:2]
```

```
[ ]: set.seed(1)
ggplot(customer_data, aes(x =Annual.Income..k.., y = Spending.Score..1.100.)) +
  geom_point(stat = "identity", aes(color = as.factor(k6$cluster))) +
  scale_color_discrete(name=" ",
              breaks=c("1", "2", "3", "4", "5","6"),
              labels=c("Cluster 1", "Cluster 2", "Cluster 3", "Cluster 4",␣
↪"Cluster 5","Cluster 6")) +
 ggtitle("Segments of Mall Customers", subtitle = "Using K-means Clustering")
```

10.2 From the above visualization, we observe that there is a distribution of 6 clusters as follows –

Cluster 6 and 4 – These clusters represent the customer_data with the medium income salary as well as the medium annual spend of salary.

Cluster 1 – This cluster represents the customer_data having a high annual income as well as a high annual spend.

Cluster 3 – This cluster denotes the customer_data with low annual income as well as low yearly spend of income.

Cluster 2 – This cluster denotes a high annual income and low yearly spend.

Cluster 5 – This cluster represents a low annual income but its high yearly expenditure.

[ ]:
```
kCols=function(vec){cols=rainbow (length (unique (vec)))
return (cols[as.numeric(as.factor(vec))])}

digCluster<-k6$cluster; dignm<-as.character(digCluster); # K-means clusters

plot(pcclust$x[,1:2], col =kCols(digCluster),pch =19,xlab␣
 ↪="K-means",ylab="classes")
legend("bottomleft",unique(dignm),fill=unique(kCols(digCluster)))
```

CONCLUSION

With the help of clustering, we can understand the variables much better, prompting us to take careful decisions. With the identification of customers, companies can release products and services that target customers based on several parameters like income, age, spending patterns, etc. Furthermore, more complex patterns like product reviews are taken into consideration for better segmentation.

REFERENCES

DATASET TAKEN FROM

https://www.kaggle.com/vjchoudhary7/customer-segmentation-tutorial-in-python

CLUSTERING IDEAS

1. Determining The Optimal Number Of Clusters: 3 Must Know Methods – Datanovia
2. The complete guide to clustering analysis: k-means and hierarchical clustering by hand and in R - Stats and R
3. https://towardsdatascience.com/cheat-sheet-to-implementing-7-methods-for-selecting-optimal-number-of-clusters-in-python-898241e1d6ad